# Java Virtual Machine Internals

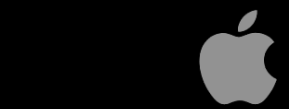
## Session 406

# Java Virtual Machine Internals

**Blaine Garst**
**Manager, Java Runtime Technologies**

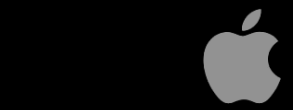
# What You'll Hear About

- Apple's VM: Internals overview
- Mac OS X 10.0 through Jaguar
- Performance advice
- What's ahead with Java 1.4
- Demo
- Q&A



# Java VM Responsibilities

- Bytecode Execution Engine
- Threads and OS interaction
- Memory Management
- Native “C” code interactions



# Apple's JVM: Best of Breed?

- Advanced language integration
- Enhanced HotSpot JVM
  - Tight OS integration
  - Superb CPU utilization
  - Best memory utilization

# Language Integration

- "C" integration with JDirect
  - Program "C" APIs in Java
  - JNI stubs built on-the-fly
- "ObjC" integration with JavaBridge
  - Cocoa framework mapped 1:1
  - JNI stubs pre-built

# Java Virtual Machine

- Use HotSpot VM with Client compiler
  - 10% time in on-the-fly built interpreter
  - 90% time in JIT compiled code
- Patented low-cost synchronization
- State-of-the-art Generational GC

# Tight OS Integration

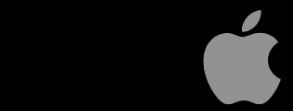
- Natural fit to Darwin
    - Threads are 1:1 with Mach
    - Files are 1:1 with BSD
    - Network is 1:1 with BSD
- Use Mach exceptions for NPE
- Abandon dirty pages

# Superb CPU Utilization

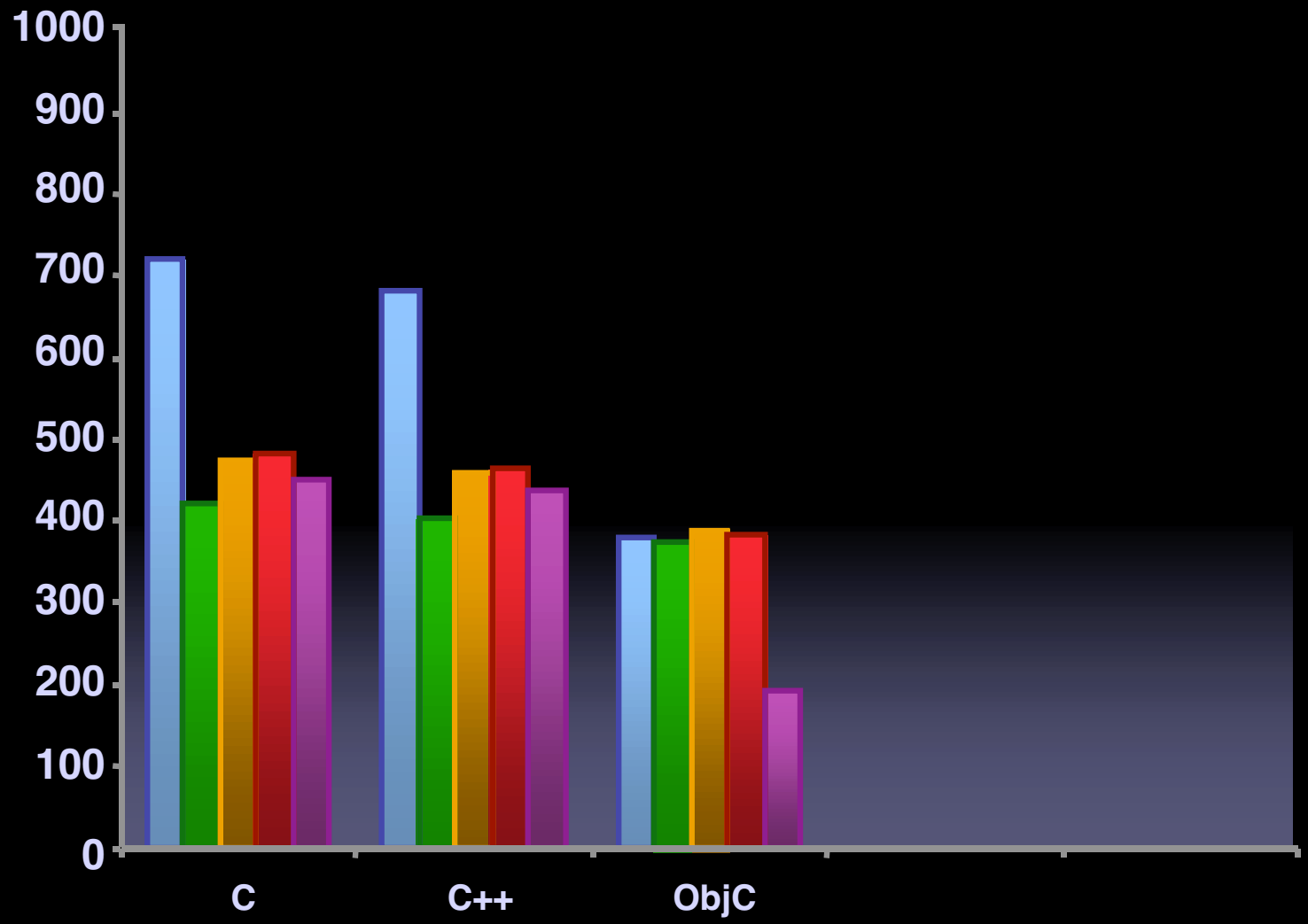
- Codegen varies for MP and G4
- Use Velocity Engine for fast copying
- Use private ABI between compiled methods
- Cached Locals
- Direct access to CPU timer
- “Millicode” routines

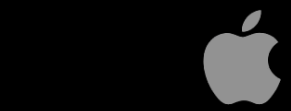
# Allocation Microbenchmark

- Threads in allocate *n*, free *n* race
- Compare C, C++, ObjC, Java
- Run it on dual MP
- Note: about Microbenchmarks
    - Contrived usage pattern
    - Your mileage will be way less

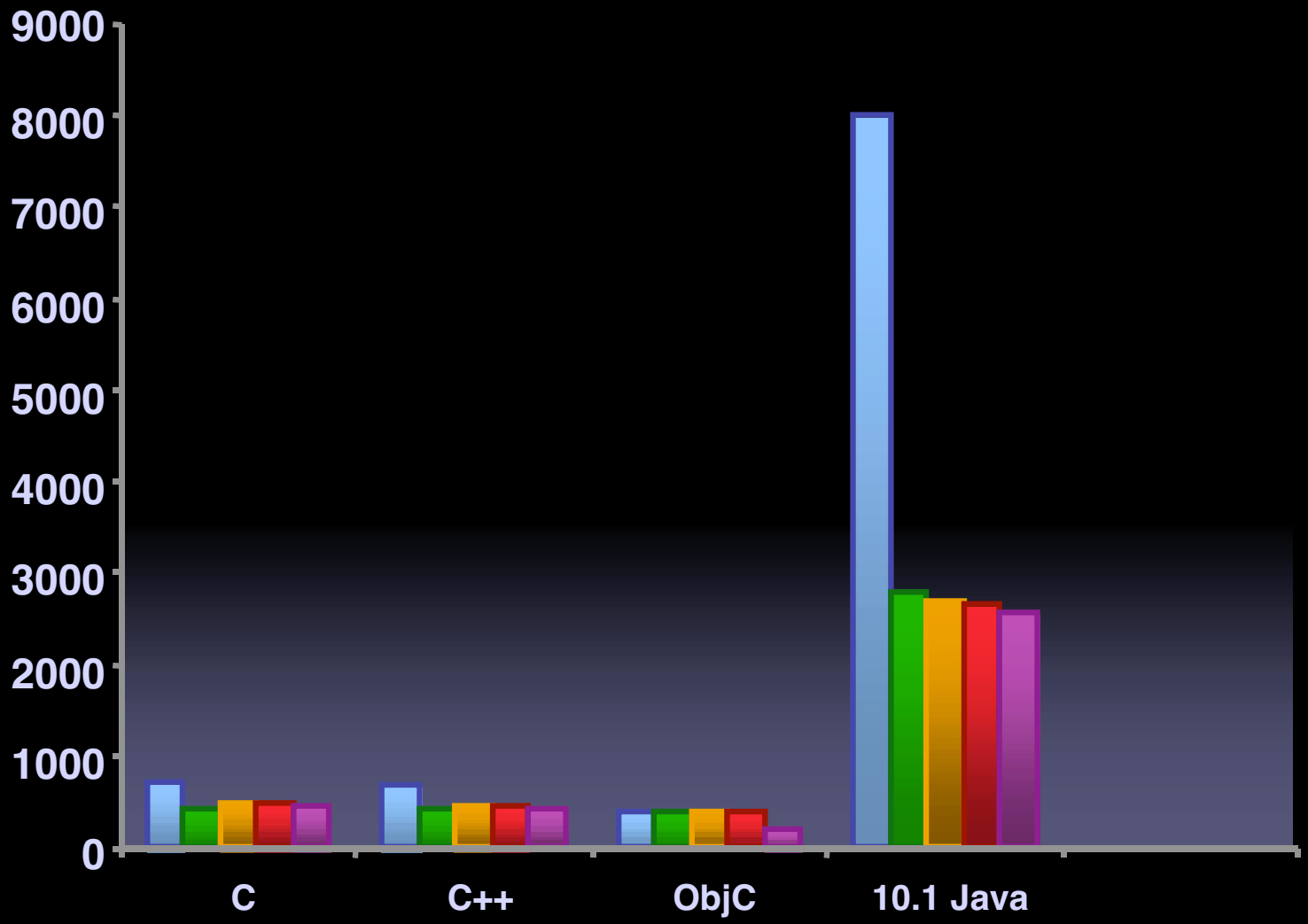


# Allocation MB Results

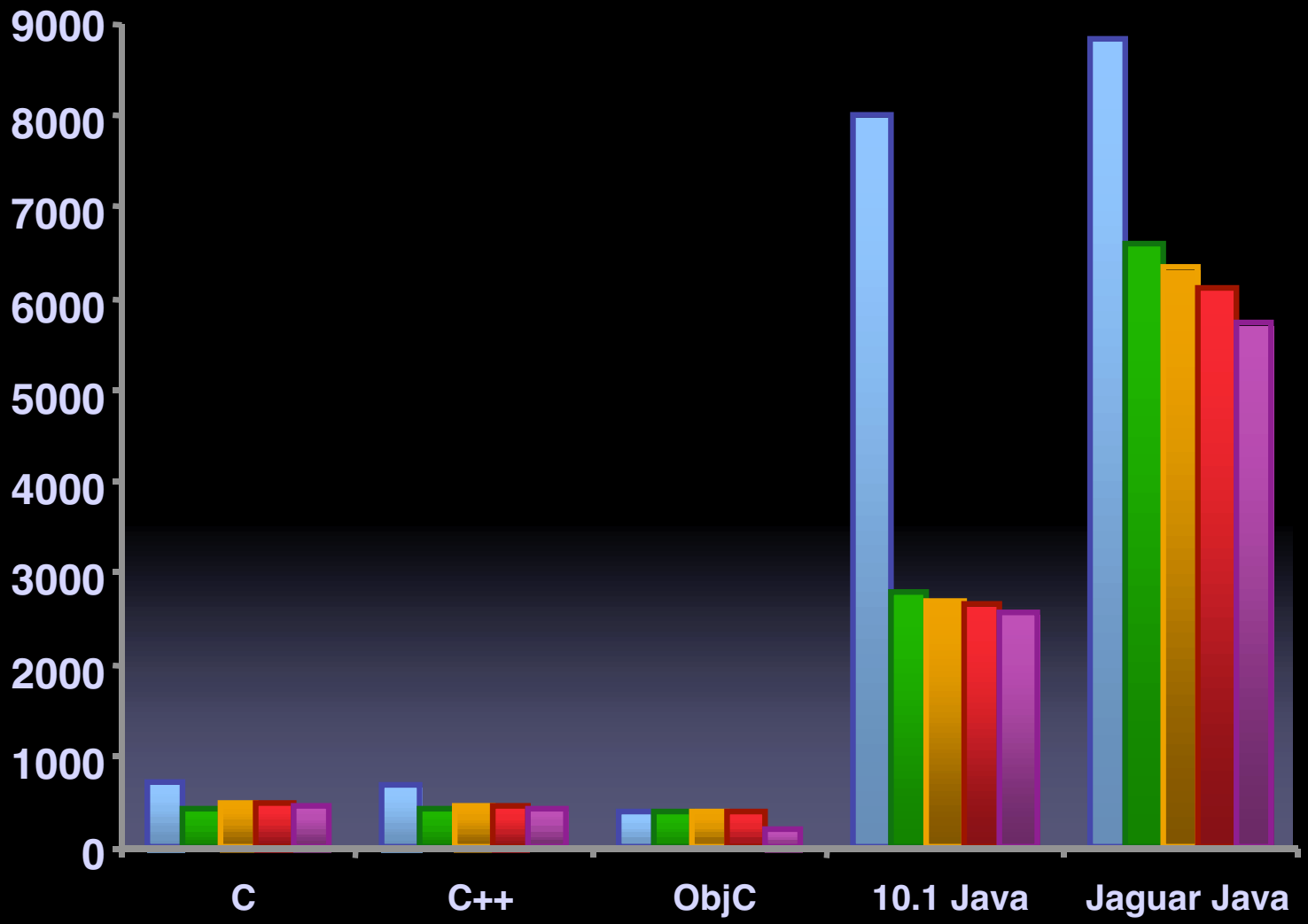


# Allocation MB Results



# Allocation MB Results



# Memory Utilization

- Pre Mac OS X 10.0
  - Jar files != shared libraries
  - Every app
    - Reads System Jars
    - Processes System Jars
    - Has SWING bytecodes copy
- Mac OS X 10.0
  - Shared Generation!

# Sun's HotSpot GC

| | | |
|---|---|---|
| Eden | ptr = mem<br>mem += size | Baby |
| New | 2-Space Copy | Child |
| Tenured | Mark & Sweep<br>or Train | Adult |
| Permanent | Mark & Sweep | Seniors |

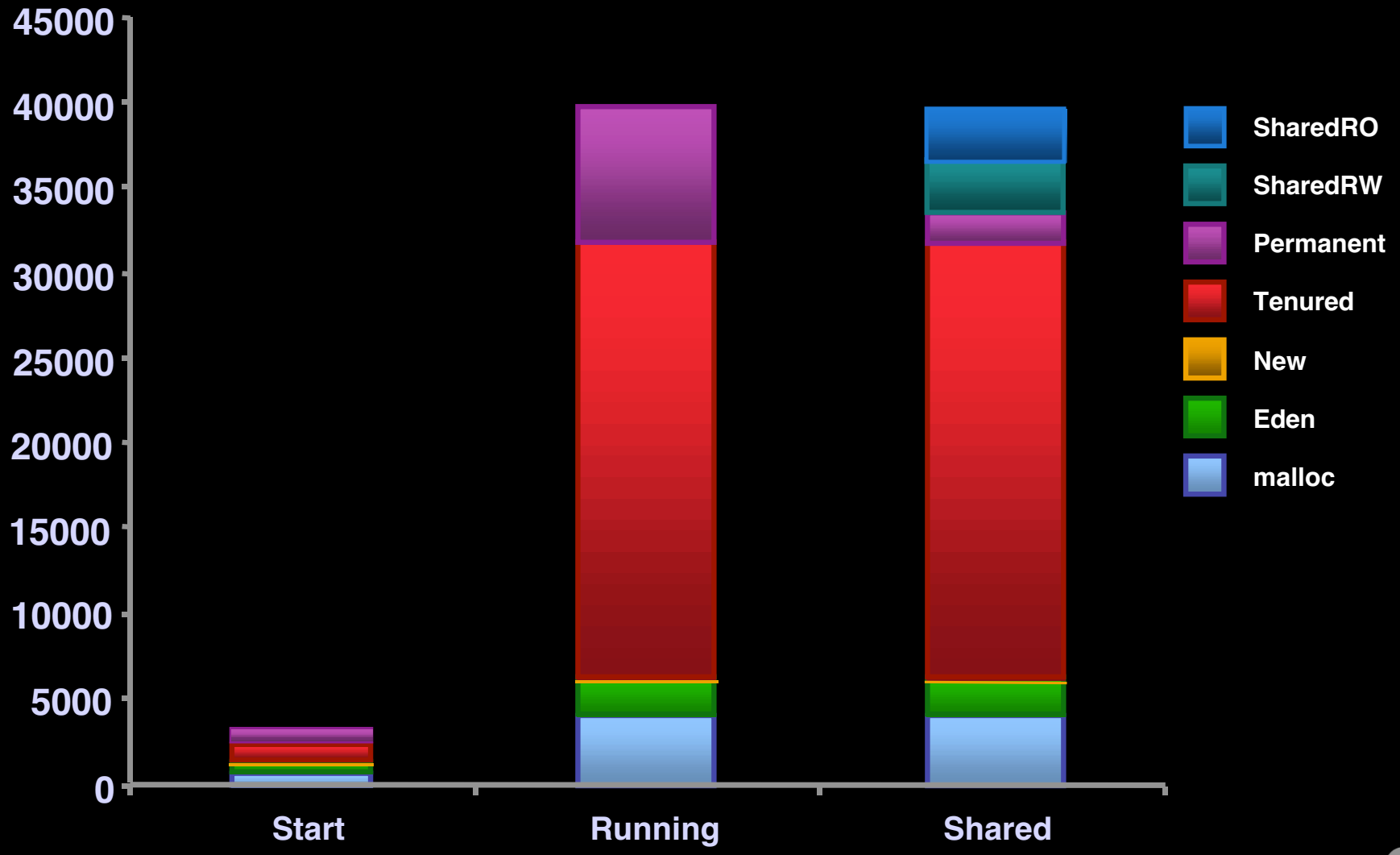
# Shared Generation Idea

- Observe: meta-data objects never die
- Extend: add “immortal” generation
- Pre-process System Jars once
    - “Building Java Shared Archive”
- Subsequently mmap, patch, and go!

# Apple's HotSpot GC

| | | |
|---|---|---|
| Eden | ptr = mem<br>mem += size | Baby |
| New | 2-Space Copy | Child |
| Tenured | Mark & Sweep<br>or Train | Adult |
| Permanent | Mark & Sweep | Seniors |
| Shared | None! | Immortal |

# Memory Savings



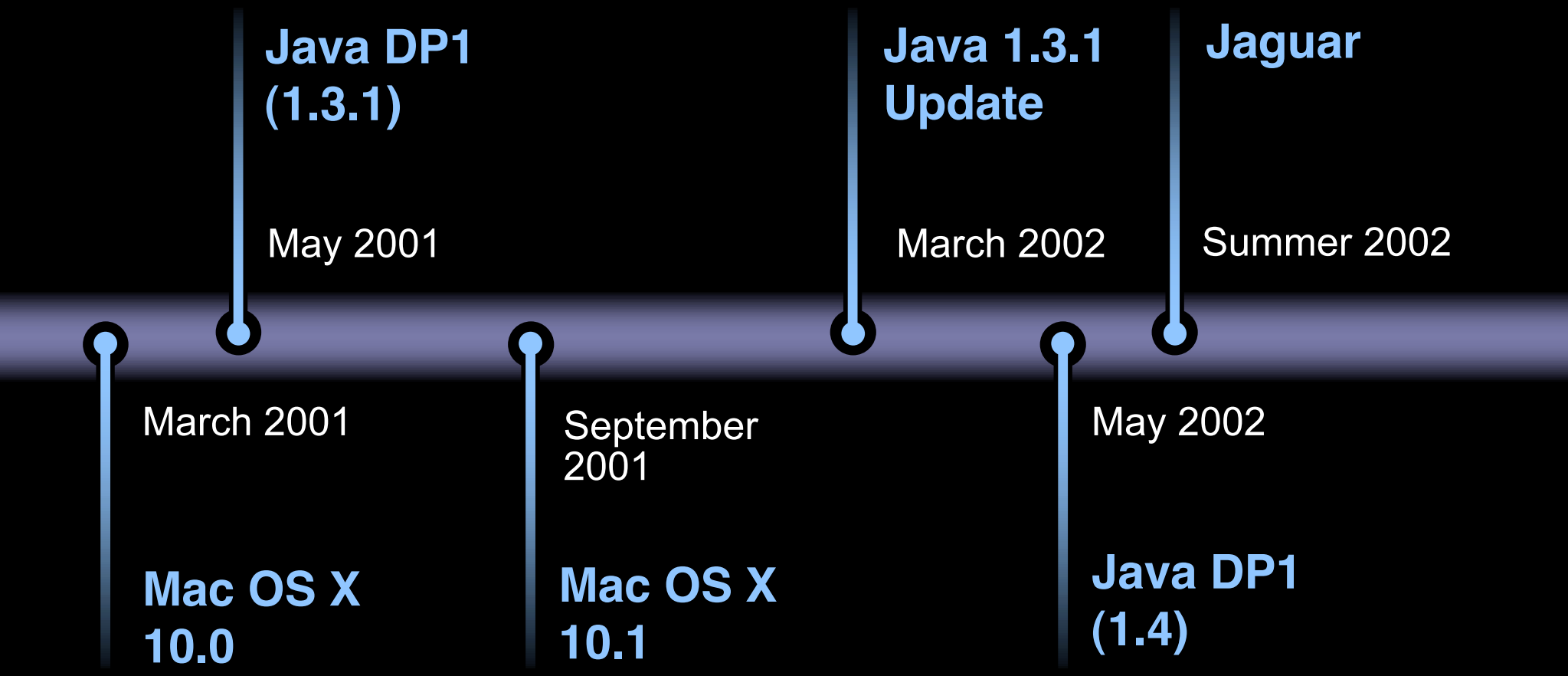
# Shared Gen: Benefits

- Less I/O for Jar file
- Less CPU for Jar file processing
- Less CPU for Garbage Collection
- Better packing of memory

# Sun's HotSpot GC***

| Eden | ptr = mem<br>mem += size | Baby |
|---|---|---|
| New | 2-Space Copy | Child |
| Tenured | Mark & Sweep<br>or Train | Adult |
| Permanent | Mark & Sweep | Seniors |
| Shared | None! | Immortal |

# Timeline


March 2001
Mac OS X 10.0
May 2001
Java DP1 (1.3.1)
September 2001
Mac OS X 10.1
March 2002
Java 1.3.1 Update
May 2002
Java DP1 (1.4)
Summer 2002
Jaguar

# Mac OS X 10.1 *September 2001*

- Java 1.3.1
- Practical debugging
- Practical profiling
- Velocity Engine copying
- Cached Locals
- Shared Generation improvements

# Java 1.3.1 SU *March 2002*

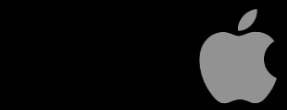
- Extend Shared Generation to constant pools
  - More Read-only memory
  - Less GC overhead
- Thread Local Eden (java -XX:+UseTLE)
- java -server

# Jaguar

- Multiple JDK capability
- Two-level Namespace
    - jnilibs can be dylibs
- Support for 1.5G Heap (java -mx1500M)
- Improved synchronization (*eieio*)
- Updated to 1.3.1_03
- (java -incgc ignored)

# Performance Tips

- Go to 407 Java Performance talk!
- Use Borland's Optimizeit
- Use Hewlett-Packard's hpprof
- Do it yourself: JVMPI
- Use built-in profilers



# Built-in Profiling Support

- Basic CPU and Monitor profiling
  - java -Xrunhprof:cpu=sample MyApp
  - java -Xrunhprof:monitor=y MyApp
- Allocation profiling
  - java -Xaprof MyApp
- Single Thread profiling
  - java -Xprof MyApp

# Beyond Jaguar

# New 1.4 Packages

java.nio, java.nio.channels, java.nio.charset

javax.xml

javax.security

org.apache.crimson, org.apache.xalan, org.apache.xml, org.apache.xpath

org.w3c.dom

org.xml.sax

java.util.prefs, java.util.logging, java.util.regex

# *Java in a Nutshell* Top Ten

www.onjava.com/pub/a/onjava/2002/03/06/topten.html

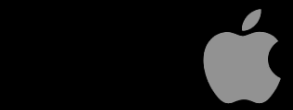
- Parsing, Transforming XML
- Preferences, Logging
- SSLSocket, LinkedHashMap
- Memory Mapped files
- Non-blocking I/O
- Regular expressions
- Language assertions

# Mac OS X J2SE 1.4 Developer Preview

- Contains HotSpot 1.4 Client Compiler
- Contains 1.4 classes.jar
- Contains 1.3.1 ui.jar
- Get it

  developer.apple.com/java
- Use it

  /usr/local/bin/juse 1.4

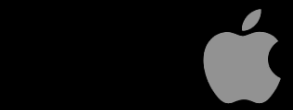


# Java 1.4 VM: Compiler Overview

- "Deoptimization": toss running compiled code
  - Allows full-speed debugging
  - Allows inlining of non-final methods
- Low-level Intermediate Representation
  - Easier/better register allocation
  - Better fit to PPC instructions
  - Peephole optimizations easier

# Java 1.4: New APIs

- Native Buffers
  - Provides external storage for arrays of bits
  - Compiler intrinsics eliminate JNI costs
- New I/O
  - IPv6
  - Non-blocking I/O
- Preferences

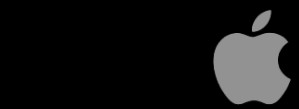


# Demo

**Greg Parker**

# Wrap Up

- Apple Java VM: Best of Breed
- Even better in Jaguar
- Try out JDK 1.4 VM
- Still to come
    - JDK 1.4 GUI
    - Even better Shared Generation
    - Compiler tuning



# Roadmap

| | |
|---|---|
| **FF003 Java:**<br>What do you think? | Room J1<br>**Wed., 5:00pm** |
| **407 Java Performance:**<br>JDK 1.4 Compiler and NIO | Room C<br>**Fri., 9:00am** |

# Who to Contact

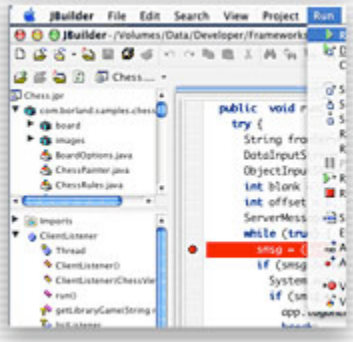
**Blaine Garst**
Manager, Java Runtime Technology
**blaine@apple.com**

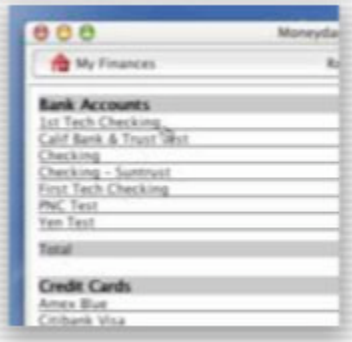
**Greg Parker**
Java Runtime Engineer
**gparker@apple.com**

**Alan Samuel**
Java Technologies Evangelist
**bluker1@apple.com**

**http://developer.apple.com/wwdc2002/urls.html**

# Q&A






**Alan Samuel**
**Java Technologies Evangelist**
**bluker1@apple.com**


**http://developer.apple.com/wwdc2002/urls.html**


WWDC2002


WWDC2002


WWDC2002